**IoT embedded systems lie at the heart of the Industry 4.0 revolution. They are the backbone of smart factories, where data gathered from interconnected devices, machines, and systems allows manufacturers to optimize operational efficiency, streamline supply chains, ensure quality control and foster innovation.**

**In this article, we’ll explore what IoT embedded systems are, how they work, and how they can help you optimize the processes at your manufacturing company.**

**Introduction to IoT and embedded systems**

The Internet of Things (IoT) and embedded systems are two interconnected technologies that have revolutionized the way we live and work. IoT refers to a network of physical devices embedded with sensors, software, and connectivity, enabling them to collect and exchange data. Embedded systems, on the other hand, are specialized computer systems designed to perform dedicated functions within larger systems. Together, they form the backbone of modern smart technologies, driving innovation across various industries.

**What is IoT?**

IoT, or the Internet of Things, is a vast network of physical devices, vehicles, home appliances, and other items embedded with sensors, software, and connectivity. This network allows these devices to collect and exchange data with other devices and systems over the internet. IoT devices can range from simple sensors that monitor environmental conditions to complex systems that manage entire industrial processes. They are used in a wide array of industries, including healthcare, manufacturing, transportation, and smart homes, transforming how we interact with the world around us.

**What is an embedded system?**

An embedded system is a small computer system designed to perform a specific task or set of tasks. It typically consists of a combination of hardware and software components that work together to provide a particular function or service. Embedded systems are ubiquitous, found in consumer electronics like smartphones and smartwatches, industrial control systems, medical devices, and automotive systems. These systems are integral to the operation of many modern technologies, providing the necessary control and processing capabilities.

**Characteristics of IoT embedded systems**

Traditionally, embedded systems are a combination of hardware and software that perform a dedicated function and typically control the operations of a machine it’s embedded within. The hardware elements include microcontrollers or microprocessors, actuators and sensors. The software components include an operating system, device drivers, and application software.

The Internet revolutionized the way we interact with technology. Today, embedded systems also include communication models, like wi-fi, Bluetooth or cellular connectivity, which allow them to collect and exchange data over the Internet that can be used later for monitoring and various analyses. **Embedded systems that are used in IoT devices are called IoT embedded systems.**

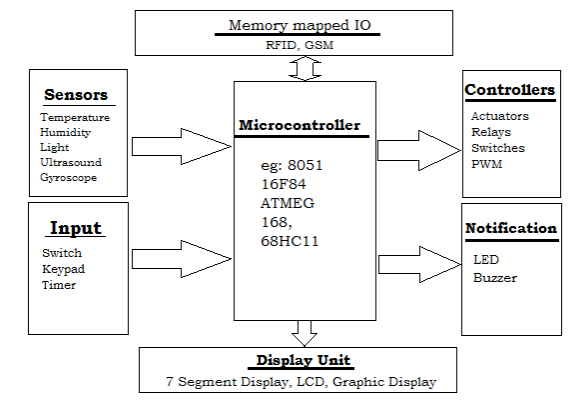
IoT embedded systems possess several distinctive characteristics that set them apart from traditional embedded systems:

* **Connectivity**: IoT embedded systems are designed to connect to the internet or other networks, enabling communication with other devices and systems. This connectivity is crucial for data exchange and remote monitoring.
* **Sensing and actuation**: These systems often include sensors that detect changes in the physical environment and actuators that perform actions based on sensor data. This capability allows IoT devices to interact dynamically with their surroundings.
* **Data processing**: IoT embedded systems can process data in real-time, making decisions and taking actions based on the data they collect. This real-time processing is essential for applications that require immediate responses.
* **Energy efficiency**: Given that many IoT devices operate on batteries or other limited power sources, energy efficiency is a critical design consideration. IoT embedded systems are optimised to consume minimal power while maintaining functionality.

**How do IoT embedded systems work?**

IoT embedded systems enable devices to become part of the Internet-of-Things ecosystem. They are used widely in smart homes, healthcare, automotive and especially in the Industry 4.0 sector.

The sensors, which are part of the embedded system, gather information about physical surroundings, for example, temperature, vibrations or humidity. The collected data is then exchanged with other devices or systems over the Internet. The data can then be analysed and used to make more informed decisions, automate processes or provide real-time feedback on the state of a device.



A microcontroller is a compact integrated circuit designed to govern a specific operation in an embedded system. A typical microcontroller includes a processor, memory and input/output (I/O) peripherals on a single chip.

Sometimes referred to as an *embedded controller* or *microcontroller unit* (MCU), microcontrollers are found in automobile engine control systems, robots, office machines, medical devices, mobile radio transceivers, vending machines and home appliances, among other devices. They're simple miniature PCs designed to control small features of a larger component without a complex front-end operating system.

**How do microcontrollers work?**

A microcontroller is embedded inside of a system to control a single function in a device. It uses its central processor to interpret data it receives from its I/O peripherals. The information that the microcontroller receives is temporarily stored in its data memory, where the processor accesses it and uses instructions stored in its program memory to decipher and apply the incoming data. It then uses its I/O peripherals to communicate and take the appropriate action.

Microcontrollers are used in an array of systems and devices. Devices often use multiple microcontrollers that work together in the device to handle their respective tasks.

For example, a car has many microcontrollers that control various individual systems, such as the antilock braking system, traction control, fuel injection and suspension control. Each microcontroller communicates with the others to inform them of the correct actions. Some might communicate with a more complex central computer within the car, and others might only communicate with other microcontrollers. They send and receive data using their I/O peripherals and process that data to perform their designated tasks.

**What are the elements of a microcontroller?**

The core elements that make up a microcontroller are the central processing unit (CPU), memory and I/O peripherals.

**CPU**

Also known as a *processor*, a CPU is the brain of the device. It processes and responds to various instructions that direct the microcontroller's function. This involves performing basic arithmetic, logic and I/O operations. It also performs data transfer operations, which communicate commands to other components in the larger embedded system.

**Memory**

A microcontroller's memory stores the data that the processor receives and uses to respond to instructions it's programmed to carry out. A microcontroller has two main memory types:

1. **Program memory.** This stores long-term information about the instructions that the CPU carries out. Program memory is non-volatile memory, meaning it stores information over time without needing a power supply.
2. **Data memory.** This temporary data storage is used while the instructions are being executed. Data memory is volatile, meaning the data it holds is temporary and is only maintained if the device is connected to a power source.

**I/O peripherals**

The I/O devices are the interface for the processor to the outside world. The input ports receive information and send it to the processor in the form of binary data. The processor receives that data and sends the necessary instructions to output devices, which execute tasks external to the microcontroller.

**Other elements**

While the processor, memory and I/O peripherals are the defining elements of the microprocessor, there are other elements that are frequently included. The term *I/O peripheral* refers to a supporting component that interfaces with the memory and processor. There are many supporting components that can be classified as peripherals. Having some manifestation of an I/O peripheral is elemental to a microprocessor because it is the mechanism through which the processor functions.

Other supporting elements of a microcontroller include the following:

* **Analog-to-digital converter.** An ADC is a circuit that converts analog signals to digital signals. It lets the processor at the center of the microcontroller interface with external analog devices, such as sensors.
* **Digital-to-analog converter.** A DAC performs the inverse function of an ADC, letting the microcontroller's processor communicate its outgoing signals to external analog components.
* **System bus.** The system bus is the connective wire that links together all components of the microcontroller.
* **Serial port.** The serial port is one example of an I/O port that enables the microcontroller to connect to external components. It has a similar function to a USB or a parallel port but differs in the way it exchanges bits.

A diagram of a microcontroller component

AI-generated content may be incorrect.

**Microcontroller features**

Microcontroller processors vary based on the application. Options range from the simple 4-bit, 8-bit or 16-bit processors to more complex 32-bit or 64-bit processors. Microcontrollers can use volatile memory, such as RAM, and non-volatile memory types, including flash memory, erasable programmable read-only memory and electrically erasable programmable ROM.

Generally, microcontrollers are usable without additional computing components. They're designed with sufficient onboard memory, as well as offering pins for general I/O operations, so they can directly interface with sensors and other components.

Microcontroller architecture is based on the Harvard architecture or Von Neumann architecture. They offer different methods of exchanging data between the processor and memory. With Harvard architecture, the data bus and instruction are separate, enabling simultaneous transfers. With a Von Neumann architecture, one bus is used for both data and instructions.

Microcontroller processors are based on complex instruction set computer (CISC) or reduced instruction set computer (RISC). CISC generally has around 80 instructions, while RISC has about 30. CISC also has more addressing modes, 12 to 24 compared to RISC's three to five.

CISC is easier to implement and uses memory more efficiently, but it can have performance degradation because of the higher number of clock cycles needed to execute instructions. RISC places more emphasis on software and provides better performance than CISC processors, which emphasize hardware. CISC has a simplified instruction set and, therefore, increased design simplicity. However, because of the emphasis RISC places on software, the software can be more complex. Which one is used depends on the application.

When they first became available, microcontrollers only used assembly language. Today, the [C](https://www.techtarget.com/searchwindowsserver/definition/C) programming language is a popular option. Python and JavaScript are also common microprocessor languages.

MCUs feature I/O pins to implement peripheral functions, such as ADCs, liquid-crystal display controllers, real-time clocks, universal synchronous/asynchronous receiver-transmitters, timers, universal asynchronous receiver-transmitters and USB connectivity. Internet of things (IoT) sensors that gather data such as humidity and temperature are also often attached to microcontrollers.

**Types of microcontrollers**

Microcontrollers can be classified according to data size and architecture. Common types include the following:

* **8-bit microcontroller.**These MCUs can only transmit 8 bits of data at a given time. However, they consume less power compared to larger data sizes.
* **16-bit** **microcontroller.** These microcontrollers have higher [clock speeds](https://www.techtarget.com/whatis/definition/clock-speed) and more memory than 8-bit microcontrollers. They are two times faster than 8-bit microcontrollers.
* **32-bit microcontroller.**These high-speed microcontrollers are faster and have more processing capacity than 16-bit ones. However, their power consumption is significantly higher.

Microcontrollers use one of two architectures:

1. **Von Neumann architecture microcontrollers**perform one operation at a time because there is only one internal bus to handle both memory and data.
2. **Harvard architecture** **microcontrollers** provide high performance compared with Von Neumann ones. This is because they have separate buses for processing instructions and moving data.

Examples of microcontroller models include the following:

* **MCS-51.** Intel developed this single-chip microcontroller type in 1980. It is also referred to as an *8051 microcontroller*. It used CISC and the Harvard architecture and came in 8-, 16- and 32-bit data sizes. Intel stopped making MCS-51 in the early 2000s, though other chipmakers offer enhanced versions of it.
* **AVR.**Atmel developed this 8-bit single-chip RISC microcontroller in 1996, using a modified Harvard architecture. It became a family of microcontrollers that was one of the first to use on-chip flash computer memory to provide program storage. Microchip Technology acquired Atmel in 2016 and continues to make AVR microcontrollers.
* **Programmable Intelligent Computer.**General Instrument developed the PIC microcontroller in 1976 under the name Programmable Interface Controller. This family of microcontrollers can be programmed to carry out different tasks, such as controlling electrical processes in homes, vehicles and medical facilities.
* **Advanced RISC Machines.** [Arm](https://www.techtarget.com/whatis/definition/ARM-processor) microcontrollers are also known as *Arm Cortex-M microcontrollers*. These lightweight microcontrollers are used in mobile electronic devices, as well as in manufacturing settings.They are designed to be energy-efficient and suitable for a range of embedded systems. These microcontrollers are part of the Arm family of processors that Acorn Computers developed in the early 1980s.

IoT embedded systems bring many benefits to modern manufacturing. Let’s take a look at a couple of the most common applications.



**Hardware and software for IoT embedded systems**

IoT embedded systems are composed of both hardware and software components, each playing a vital role in their operation.

* **Microcontrollers**: These small computers are designed to perform specific tasks, often controlling sensors and actuators within the IoT embedded system.
* **Sensors**: Devices that detect changes in the physical world, such as temperature, humidity, or motion, providing the data needed for the system to function.
* **Actuators**: Devices that take action based on sensor data, such as turning on a light or activating a motor, enabling the system to interact with its environment.
* **Communication modules**: Components that allow IoT embedded systems to communicate with other devices and systems over the internet, facilitating data exchange and remote control.
* **Power management**: Systems that manage the power consumption of the IoT embedded system, ensuring efficient use of batteries or other power sources.

**How are IoT embedded systems used in Industry 4.0?**

**Predictive maintenance**

IoT embedded systems and embedded devices can be used to collect real-time data, such as temperature, pressure or vibration measurements, from machines at the manufacturing plant. Then, the data can be analyzed using machine learning algorithms. The insights can help predict when maintenance is required, preventing unexpected downtime. It can also help minimize maintenance costs since the maintenance is done only when needed.

**Quality control**

IoT embedded systems, powered by embedded software, are often used to monitor the production line in real-time in order to detect potential defects in manufactured products. As a result, defects can be spotted quickly and corrected immediately, which means reduced waste and better product quality.

**Smart logistics**

Chain efficiency is a key factor for manufacturers. IoT systems can be used to track inventory levels and automate the ordering process. As a result, there’s a lower risk of stockouts or overstocking. What is more, IoT embedded systems can also be used to track the delivery of materials in a supply chain. Manufacturers can then track their inventory levels in real-time, improving supply chain efficiency and minimizing delays.

**Enabling remote monitoring and management**

By gathering data from sensors, for example, thermostats, an IoT operating system in IoT embedded systems can monitor the performance and condition of machinery and equipment remotely. Real-time data can help identify potential problems and reduce the risk of failure by taking preventive actions on time. Also, the insights can help schedule maintenance activities based on the actual wear of the equipment, leading to increased device lifespan.

**Production optimization**

Through collecting and analyzing data from various points of the production process, such as machine parameters or worker input, IoT embedded systems, combined with innovative software development, can help identify possible bottlenecks or inefficiencies. Manufacturers can then make quick adjustments to optimize the production process.